This is the numerics library, common to all numeric subroutines.

```
ns: nm
```

Number of iterations maximally performed in an algorithm to avoid endless loops, if the procedure converges too slowly.

```
1000 var, max-iterations
```

By default, we do not show all the steps performed.

```
false var, num-debug
```

Width of the exponent display. Your numbers will only look nice, if your largest exponent is smaller than this.

```
2 var, exp-size
```

Width w of the mantissa. We have three places in front, the sign, a number and the decimal point, so we have $w - 3$ decimal places.

```
var mantissa-size \ n --
```

Set mantissa size.

```
: mantissa! \ n --
    dup mantissa-size ! 3 - ## ;
```

The default is ± 1.0000000.

```
10 mantissa!
```

This is the error bound we normally use, the default is $\epsilon = 10^{-16}$.

```
var epsilon
: eps \ -- n
   epsilon @ ;
: eps! \ n --
   epsilon ! ;
1e-16 eps!
```

In a table, we usually print every line. If you want only every n-th line, set this to n. The first and last line will always be shown.

```
1 var, print-lines
```

We use machine numbers by default. If you want big floats, which are considerabely slower, you may set e.g. 30 bigfloats. If you need the precision, but don't want to display all the places, set **n#** to a convenient value. To avoid problems with rounding or accuracy, you should use **big-floats** once and not change it thereafter.

```
: short-floats \ --
   19 mantissa! 1e-16 eps! ;
```

```
: big-floats \ n --
   dup n:1+ dup n# mantissa! F10 swap n:neg n:^ eps! ;
```

```
short-floats
```

Forward declaration of symbols, we often use and like to change.

```
defer: f
```

Return the decimal log of n. Right now ln does not work for bfloat, so we convert to a float. Of course, we don't have the precision we would expect. Will be fixed in 20.06, hopefully.

```
: ld \ n -- n
   n:float n:ln 10 n:ln n:/ ;
```

Return the exponent of n. For $n > 300$ or so, this is not correct because of the ln error. We could divide successively by 10 as a workaround, if the fix is not made soon.

```
: exponent \ n -- n
   dup 0 > if ld  else
   dup 0 < if n:neg ld else
   then then n:floor n:int ;
```

Return the length of the exponent.

```
: exp-length \ n -- n
   dup 0 = if drop 1 else
   dup 0 < if n:neg then
   n:1+ ld n:ceil n:int then ;
```

Print the sign of the exponent.

```
: print-sign \ n -- n
   dup 0 < if "-" . else "+" . then ;
```

Print zeroes to fill the exponent.

```
: print-zeroes \ n -- n
   dup exp-length exp-size @ swap n:- ( "0" . ) swap times ;
```

Print the value of the exponent.

```
: print-value \ n --
   n:abs . ;
```

Print the exponent as ±01.

```
: print-exponent \ n -- +|-00n
   print-sign print-zeroes print-value ;
```

Return the mantissa.

```
: mantissa \ n exp -- n
   10. swap n:^ n:/ ;
```

Print a number in the format $[-]1.0000000e \pm 01$.

```
: f. \ n --
   mantissa-size @ #>
   dup exponent dup  >r \ n exp
   mantissa . "e" .
   r> print-exponent ;
```